



# Testing APIs Postman

JEE Microservices  
@ CGS IT – 2023

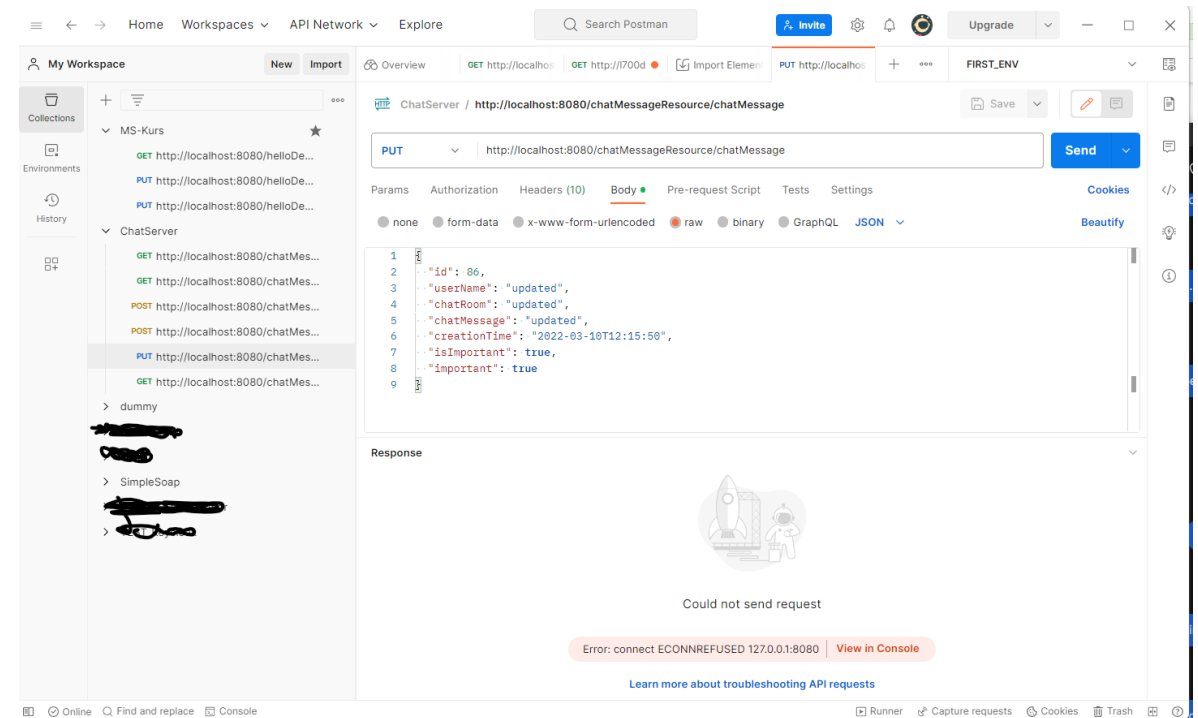
v.1.0.3

# Inhalt

- Postman Überblick
- Postman Features
- Postman Testing Request und Tests
- Postman OpenAPI Editierung und Validierung

# Postman – UI - Überblick

- Eingabe von URL, HTTP Methode
- Parameter & Header
- In den einzelnen TABS unter dem URL möglich
- Hier PUT Request zu <http://localhost:8080/chatMessageResource/chatMessage>
- In der Content Section das JSON das mitgesendet wird



# Postman Tests

- Mittels Postman Tests können Request Ergebnisse geprüft werden
- Sowohl http status
- Als auch der JSON Response kann relativ leicht überprüft werden

ChatServer / http://localhost:8080/chatMessageResource/chatMessage/10

GET http://localhost:8080/chatMessageResource/chatMessage/10

Params Authorization Headers (6) Body Pre-request Script Tests Settings

```
1  const responseJson = pm.response.json();
2  pm.response.to.have.jsonBody().not.empty;
3  });
4
5  pm.test("Property is ok", () => {
6    const jsonData = pm.response.json();
7    pm.expect(jsonData.userName).to.not.empty
8  });
9
10 pm.test("Property userName has value updated", () => {
11   const jsonData = pm.response.json();
12   pm.expect(jsonData.userName).to.eqls("updated")
13 });
14
```

Body Cookies Headers (2) Test Results (6/6)

All Passed Skipped Failed

PASS Status code is 200

PASS The response has all properties

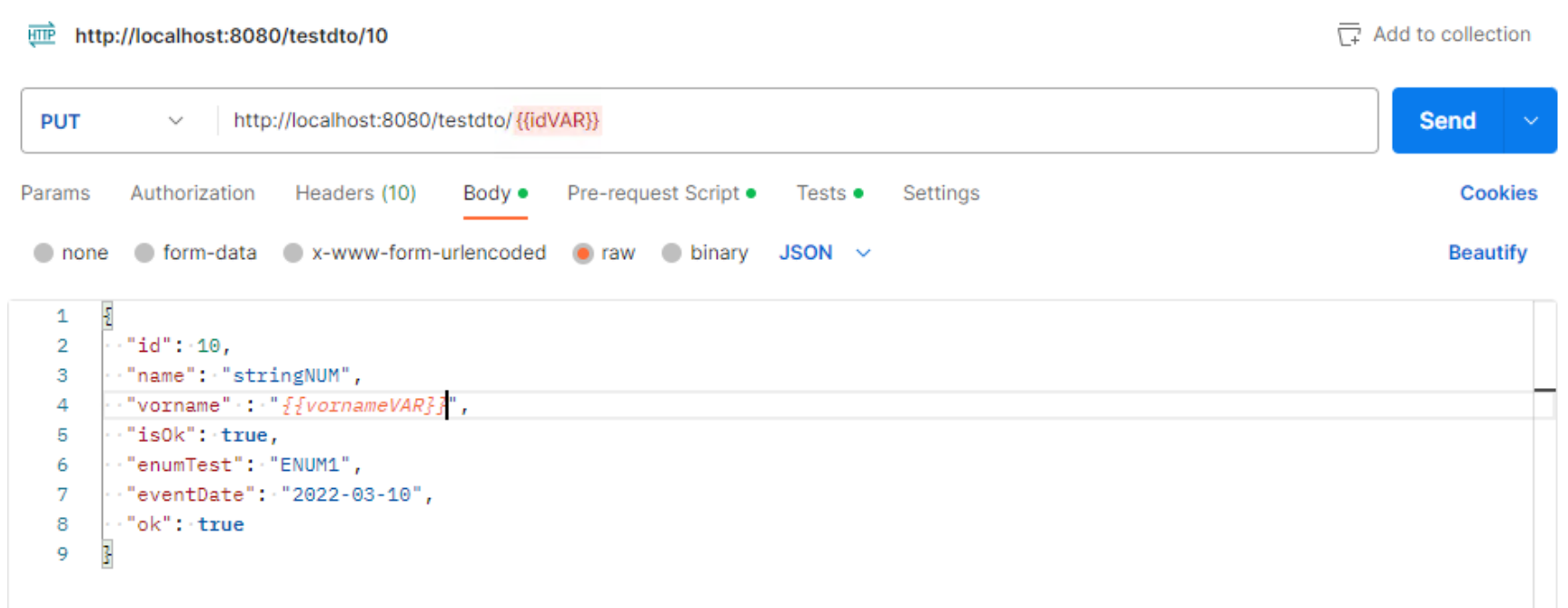
PASS Property is ok

PASS Property userName has value updated

PASS Property id is a number

PASS Property id equals 10

# Postman – Variablen in JSON



# Postman Variables

- Variablen aus File oder PRE Collection

HTTP <http://localhost:8080/testdto/10>

PUT

<http://localhost:8080/testdto/{{idVAR}}>

Params

Authorization

Headers (10)

Body ●

Pre-request Script ●

Tests ●

Settings

```
1 pm.collectionVariables.set("vornameVAR", "vornameVariableXX");  
2 pm.collectionVariables.set("idVAR", "15");
```

# Postman Tests

```
pm.test("Status code is 200", () => {  
  pm.expect(pm.response.code).to.eql(200);  
});
```

```
pm.test("The response has all properties", () => {  
  //parse the response JSON and test three properties  
  const responseJson = pm.response.json();  
  pm.response.to.have.jsonBody().not.empty;  
});
```

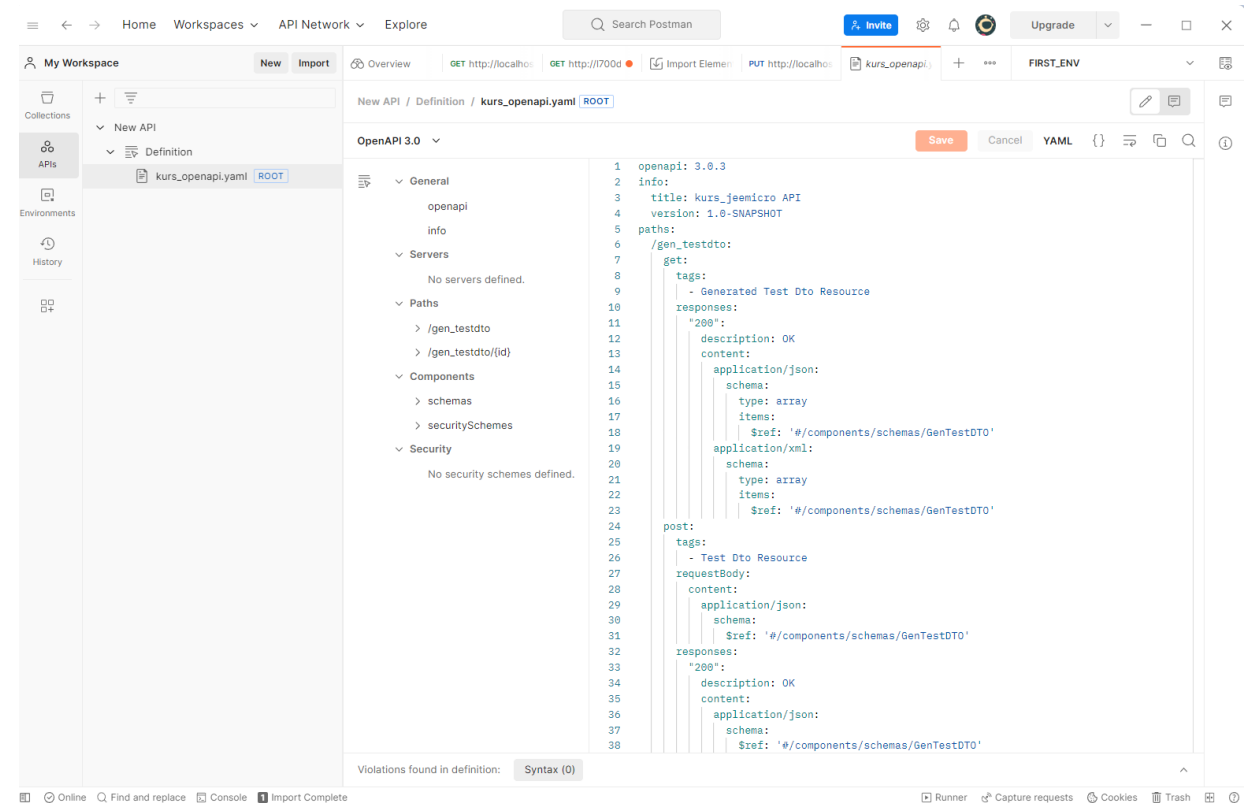
```
pm.test(" Property id equals 10 ", () => {  
  const jsonData = pm.response.json();  
  pm.expect(jsonData.id).not.null;  
  pm.expect(jsonData.id).to.eql(10);  
});
```

```
pm.test(" Property name is ok", () => {  
  const jsonData = pm.response.json();  
  pm.expect(jsonData.id).not.null;  
  pm.expect(jsonData.name).to.eql("stringNUM");  
});
```

```
pm.test(" Property name is ok", () => {  
  const jsonData = pm.response.json();  
  pm.expect(jsonData.name).to.eql("stringNUM");  
});
```

# Postman – OpenAPI - Editierung

- Postman allows to import
- Edit and validate
- As well as Export openapi Specifications





Danke für Ihre Aufmerksamkeit